ONT03854S

D.J. #3 10-17-01

Priority Papers

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application: ２００１年 ４月１１日

出願番号
Application Number: 特願２００１－１１２３５４

出願人
Applicant(s): 株式会社日立製作所

U.S. Appln. Filed 8-29-01
Inventor: H. Fujii et al
Mathingly Stanger & Malur
Docket HIT-292

２００１年 ６月１８日

特許庁長官
Commissioner,
Japan Patent Office 及川耕造

出証番号 出証特２００１－３０５７０４８

【書類名】　特許願

【整理番号】　NT00P1088

【提出日】　平成13年 4月11日

【あて先】　特許庁長官　殿

【国際特許分類】　G06F　9/30

G06F　9/31

G06F　9/32

G06F　9/45

G06F　9/455

【発明者】

【住所又は居所】　東京都国分寺市東恋ヶ窪一丁目２８０番地　株式会社日立製作所　中央研究所内

【氏名】　藤井　啓明

【発明者】

【住所又は居所】　東京都国分寺市東恋ヶ窪一丁目２８０番地　株式会社日立製作所　中央研究所内

【氏名】　田中　義一

【発明者】

【住所又は居所】　東京都国分寺市東恋ヶ窪一丁目２８０番地　株式会社日立製作所　中央研究所内

【氏名】　三木　良雄

【特許出願人】

【識別番号】　000005108

【氏名又は名称】　株式会社日立製作所

【代理人】

【識別番号】　100068504

【弁理士】

【氏名又は名称】　小川　勝男

【電話番号】　03-3661-0071

【選任した代理人】

【識別番号】　100086656

【弁理士】

【氏名又は名称】　田中　恭助

【電話番号】　03-3661-0071

【選任した代理人】

【識別番号】　100094352

【弁理士】

【氏名又は名称】　佐々木　孝

【電話番号】　03-3661-0071

【手数料の表示】

【予納台帳番号】　081423

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　動的命令変換機能を有するプロセッサシステム、該プロセッサシステムを備えたコンピュータにて実行されるバイナリートランスレーションプログラム及びそのプロセッサシステムを実装した半導体デバイス

【特許請求の範囲】

【請求項１】

異なるハードウェアプラットフォーム向けの命令バイナリコードプログラムを動的に自身の命令バイナリコードに変換しながらプログラム実行を行う動的命令変換機能を有するプロセッサシステムにおいて，前記異なるハードウェアプラットフォーム向けの命令バイナリコードプログラムから１命令づつ読み出し，該１命令づつをソフトウェアを介して解釈実行する処理流と，前記１命令づつ必要に応じて自身の命令バイナリコードに変換し，前記１命令づつ蓄積し，該蓄積された命令バイナリコード列を必要に応じて最適化する処理流を独立させ，並列に処理することを特徴とする動的命令変換機能を有するプロセッサシステム。

【請求項２】

上記命令バイナリコード列の最適化において，繰り返し処理や手続き呼出し処理などを並列に実行出来るように複数処理流を生成するように新たな命令バイナリコード列を構成することを特徴とする請求項１の動的命令変換機能を有するプロセッサシステム。

【請求項３】

上記解釈実行する処理流及び上記最適化する処理流とは別に，前記異なるハードウェアプラットフォーム向けの命令バイナリコードプログラムをキャッシュメモリに先読みする処理流を独立させ，上記解釈実行する処理流及び上記最適化する処理流と並列に処理することを特徴とする請求項１及び２の動的命令変換機能を有するプロセッサシステム。

【請求項４】

上記最適化する処理流が，所定の単位の命令バイナリコード列最適化を完了する毎に，前記最適化した命令バイナリコード列をその最適化完了時点で上記解釈実行する処理流が実行している命令コードと入れ替え，前記解釈実行する処理流

は，異なるハードウェアプラットフォーム向けの命令バイナリコードプログラムの各命令解釈実行時に前記１命令に対応する最適化変換後命令バイナリコード列が存在する場合，前記最適化変換後命令バイナリコード列を実行することを特徴とする請求項１～３の動的命令変換機能を有するプロセッサシステム。

【請求項５】

１個のＬＳＩチップに複数のマイクロプロセッサが実装されるチップマルチプロセッサとして構成され，上記複数処理流をそれぞれ異なるマイクロプロセッサで並列して同時に処理させることを特徴とする請求項１～４の動的命令変換機能を有するプロセッサシステム。

【請求項６】

１個の命令実行制御部で同時に複数の処理流を実行するように構成され，上記複数処理流を並列に実行することを特徴とする請求項１～４の動的命令変換機能を有するプロセッサシステム。

【請求項７】

上記解釈実行する処理流による実行処理中の変換後命令列と、上記最適化する処理流による前記変換後命令列の最適化後の新規変換後命令列の間で変換後命令列の入れ替え処理を行う際、排他制御を実施することを特徴とする請求項１～３の動的命令変換機能を有するプロセッサシステム。

【請求項８】

少なくとも１つの処理流から成る動的命令変換機能を有するプロセッサシステムを備え、

前記少なくとも１つの処理流は異種ハードウエアにて実行されるバイナリーコードプログラムを構成する複数の命令を順次先読みし、共有メモリに格納する処理流１と、前記共有メモリに格納された前記複数の命令を並列して同時に解釈実行する処理流２と、前記解釈実行された前記複数の命令の変換を行う処理流３から構成されることを特徴とする動的命令変換機能を有するプロセッサシステム。

【請求項９】

請求項８に記載のプロセッサシステムにおいて、前記処理流２は、前記複数の命令の内、既に変換されている命令に対し、前記変換を実行することなく前記解

釈実行を行うことを特徴とする動的命令変換機能を有するプロセッサシステム。

【請求項１０】

請求項８に記載のプロセッサシステムにおいて、前記処理流３は、前記複数の命令の内、変換されていない命令に対し該命令を変換し、該変換された該命令の並べ替え又は該変換された該命令数の削減を行うことを特徴とする動的命令変換機能を有するプロセッサシステム。

【請求項１１】

請求項８に記載のプロセッサシステムにおいて、前記処理流１、前記処理流２及び前記処理流３は互いに独立して並列に処理するように構成されることを特徴とする動的命令変換機能を有するプロセッサシステム。

【請求項１２】

少なくとも１つのマイクロプロセッサ、バス、共有メモリなどを備えた半導体デバイスにおいて、

前記少なくとも１つのマイクロプロセッサは、少なくとも１つの処理流を実行するように構成され、

前記少なくとも１つの処理流は異種ハードウエアにて実行されるバイナリーコードプログラムを構成する複数の命令を順次先読みし、前記共有メモリに格納する処理流１と、前記共有メモリに格納された前記複数の命令を並列して同時に解釈実行する処理流２と、前記解釈実行された前記複数の命令の変換を行う処理流３から構成され、

前記少なくとも１つのマイクロプロセッサは、並列して前記複数の命令を処理しうるように構成されることを特徴とする半導体デバイス。

【請求項１３】

コンピュータに複数の命令の読み出しを行う手順と、該読み出した該複数の命令の内、変換されていない命令に対し変換処理を行う手順と、該変換処理された該命令を実行する手順を並列に実行させるためのバイナリトランスレーションプログラム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は，動的命令変換機能を有するプロセッサシステムに係わり，詳しくは異なるハードウェアプラットフォーム向けの命令バイナリコードプログラムを動的に自身の命令バイナリコードに変換しながらプログラム実行を行う動的命令変換機能を有するプロセッサシステム、該プロセッサシステムを備えたコンピュータにて実行されるバイナリートランスレーションプログラム及びそのプロセッサシステムを実装した半導体デバイスに関する。

【０００２】

【従来の技術】

計算機システムを製造するメーカーは、当該計算機システムの性能向上を目的として同システムのＣＰＵ（中央処理装置）に従来と異なるアーキテクチャのマイクロプロセッサを適用する場合がある。

【０００３】

この場合に問題となるのが，従来の計算機システムとのソフトウェア互換性である。

【０００４】

基本的に，このようなアーキテクチャ変更を行った計算機システムでは，従来システムで使用していたソフトウェアが利用不能になる。

【０００５】

この問題を解決する手段として，当該ソフトウェアに対するソースコードを新規システム上でコンパイラによって再コンパイルし，当該新規システム用の命令バイナリコードを生成するという方法などが紹介されている。

【０００６】

しかし、当該新規システム使用者の手元にそのようなソースコードが無い場合には上記方法等が利用できないケースも多い。

【０００７】

上記したようなケースでも対応可能な手法として，ソフトウェア手段によって従来計算機システムにおいて使用されていたマイクロプロセッサ向けの命令を解釈実行（ｉｎｔｅｒｐｒｅｔ）する、または、当該マイクロプロセッサ向けの命

令を当該新規システムのマイクロプロセッサ向けの命令に変換（ｔｒａｎｓｌａｔｅ）して同変換後命令を直接実行するという方法がある。

【０００８】

特に従来計算機システムで使用していたソフトウェアプログラムの新規システムでの処理中に動的に後者の命令変換，変換後実行を適用する方法は動的命令変換方法（ｄｙｎａｍｉｃ　ｂｉｎａｒｙ　ｔｒａｎｓｌａｔｉｏｎ）と呼ばれ，これを実現する機能は動的命令変換機能と呼ばれる。

【０００９】

これらのソフトウェア手段については、ＩＥＥＥ学会誌ＩＥＥＥ　Ｃｏｍｐｕｔｅｒ　２０００年３月号の４０頁から４５頁に収録されている“Ｗｅｌｃｏｍｅ　ｔｏ　ｔｈｅ　Ｏｐｐｏｒｔｕｎｉｔｉｅｓ　ｏｆ　Ｂｉｎａｒｙ　Ｔｒａｎｓｌａｔｉｏｎ”という記事にて概略的に紹介されており、また、同学会誌４７頁から５２頁に収録されている“ＰＡ－ＲＩＳＣ　ｔｏ　ＩＡ－６４：　Ｔｒａｎｓｐａｒｅｎｔ　Ｅｘｅｃｕｔｉｏｎ，　Ｎｏ　Ｒｅｃｏｍｐｉｌａｔｉｏｎ”という記事は同技術の一事例を紹介している。

【００１０】

上記動的命令変換手法は，上述の計算機システムのマイクロプロセッサが変更されたようなケースへの対応としてだけでなく、あるハードウェアプラットフォームの計算機システムを利用しているユーザが、異なるハードウェアプラットフォーム上でのみ動作するソフトウェアを使用したい場合にも適用される技術である。

【００１１】

また、近年になって、動的命令変換機能を積極的にアーキテクチャに取り込んだ新規マイクロプロセッサの提案が相次ぎ，注目されている。

【００１２】

具体的な事例としては、ＩＥＥＥ学会誌ＩＥＥＥ　Ｃｏｍｐｕｔｅｒ　２０００年３月号の５４頁から５９頁に収録の“Ｄｙｎａｍｉｃ　ａｎｄ　Ｔｒａｎｓｐａｒｅｎｔ　Ｂｉｎａｒｙ　Ｔｒａｎｓｌａｔｉｏｎ”で開示されているＩＢＭ社のＢＯＡ（Ｂｉｎａｒｙ－ｔｒａｎｓｌａｔｉｏｎ　Ｏｐｔｉｍｉｚｅｄ

Ａｒｃｈｉｔｅｃｔｕｒｅ）や、Ｃａｈｎｅｒｓ ＭＩＣＲＯＰＲＯＣＥＳＳＯＲ ＲＥＰＯＲＴ Ｖｏｌｕｍｅ １４，Ａｒｃｈｉｖｅ ２の１頁および９頁から１８頁に収録の“ＴＲＡＮＳＭＥＴＡ ＢＲＥＡＫＳ Ｘ８６ ＬＯＷ－ＰＯＷＥＲ ＢＡＲＲＩＥＲ －ＶＬＩＷ Ｃｈｉｐｓ Ｕｓｅ Ｈａｒｄｗａｒｅ－Ａｓｓｉｓｔｅｄ ｘ８６ Ｅｍｕｌａｔｉｏｎ”で開示されているＴｒａｎｓｍｅｔａ社のＣｒｕｓｏｅなどが挙げられる。

【００１３】

図２に上記従来技術の動的命令変換機能を含んだ、異種ハードウェアプラットフォーム向け命令バイナリコードプログラム（以降、元命令列と略称する場合がある）実行機構の構成を示す。

【００１４】

図２において、２０１は異種ハードウェアプラットフォーム向け命令の解釈実行部，２０２は本実行機構の処理全体を制御する実行制御部、２０３は異種ハードウェアプラットフォーム向け命令列から本実行機構を有するハードウェアプラットフォームの命令列（以降、変換後命令列と略称する場合がある）を動的に生成する動的命令変換部、２０４はオペレーティングシステム（ＯＳ）が関係する処理部分等のプログラム中の特殊処理部を本実行機構を有するハードウェアプラットフォームの機能を用いてエミュレーションする特殊処理エミュレーション部、そして、２０５は本実行機構を有するプラットフォームＯＳ及びハードウェアである。

【００１５】

本実行機構による異種ハードウェアプラットフォーム向け命令バイナリコードプログラム処理がプラットフォームＯＳ及びハードウェア２０５上で起動されると、実行制御部２０２が処理を開始する。実行制御部２０２は，プログラム処理中に適時、解釈実行部２０１、動的命令変換部２０３、及び特殊処理エミュレーション部２０４に処理を依頼する。特殊処理エミュレーション部２０４は、プラットフォームＯＳ及びハードウェア２０５の機能を直接使用して依頼された処理を遂行する。

【００１６】

次に図２に係わるより詳細な処理フローに関して、図３を用いて説明する。

【００１７】

図２の実行機構が処理を開始すると、実行制御部２０２が動作を始め、処理３０１のとおり、元命令列アドレスに基づき、元命令列中の命令が参照され、当該命令に対する実行回数計測カウンタがインクリメントされる。実行回数計測カウンタは、元命令列管理表などのソフトウェアデータ構造体中に含まれる。

【００１８】

次に、処理３０２において、当該元命令列管理表が参照され、当該命令に対応する変換後命令列の有無がチェックされる。変換後命令列が存在する場合，変換後命令列領域３０８上の該当する変換後命令列ブロック３０６を元命令列管理表を参照して特定し、これを直接実行した後処理３０１に戻る。処理３０２において、変換後命令列が存在しない場合、当該命令の実行回数が検査される。同実行回数があらかじめ定められた閾値を超える場合、処理３０５が起動され、閾値以下の場合、処理３０４が起動される。処理３０４開始にあたって、実行制御部２０２は、解釈実行部２０１を呼び出す。解釈実行部２０１は、元命令列を順次参照し、各命令を解釈し、それぞれの動作に相当する処理をあらかじめ用意されたソフトウェア処理手順にしたがって実現する。

【００１９】

先述したとおり、当該命令がオペレーティングシステム（ＯＳ）が関係する処理部分等のプログラム中の特殊処理部に該当する場合には、解釈実行部２０１は、実行制御部２０２にその旨を報告する。実行制御部２０２は、特殊処理エミュレーション部２０４を起動し、特殊処理エミュレーション部２０４は、プラットフォームＯＳ及びハードウェア２０５の機能を用いて、当該処理を実施する。特殊処理が完了すると、制御は，特殊処理エミュレーション部２０４から実行制御部２０２を介して、解釈実行部２０１に戻る。解釈実行部２０１は、元命令列に分岐命令が出てくるまで上記処理を繰り返した後、制御を実行制御部２０２の処理３０１に戻す。

【００２０】

一方、処理３０５開始にあたって、実行制御部２０２は、動的命令変換部２０

３を呼び出す。動的命令変換部２０３は、分岐命令で区切られる一連の元命令列（ブロック）中の各命令を本実行機構を有するハードウェアプラットフォームの命令列に置換え、生成した変換後命令列を必要に応じて最適化した後、同変換後命令列を変換後命令列ブロック３０６として変換後命令列領域３０８上に保存する。

【００２１】

その後、動的命令変換部２０３は、制御を実行制御部２０２に戻し、実行制御部２０２は、当該新規に生成された変換後命令列ブロック３０６を直接実行した後処理３０１に戻る。実行制御部２０２は、以上の処理をプログラム終了まで繰り返す。なお，以上で説明した処理の分担は一例であり、異なる処理分担の例もあり得る。

【００２２】

上記処理フローは、１つの処理流で実現されている。このため、動的命令変換部２０３における命令変換及び最適化処理は元命令列実行処理にとってオーバヘッドとなり、元命令列処理性能を低下させる。

【００２３】

また、上記ＩＢＭ社のＢＯＡやＴｒａｎｓｍｅｔａ社のＣｒｕｓｏｅは、基本アーキテクチャにＶＬＩＷ（Ｖｅｒｙ　ｌｏｎｇ　Ｉｎｓｔｒｕｃｔｉｏｎ　Ｗｏｒｄ）方式を採用して、変換後命令列の命令レベル並列処理による高速処理、プロセッサ自体の高速周波数での動作、低消費電力化を実現することを狙っているが、上記の動的命令変換部２０３における命令変換及び最適化処理のオーバヘッドの削減は必ずしも十分でなく、更なる削減が望まれる。また、ＬＳＩテクノロジの将来動向を考慮すると、プロセッサの高速周波数動作や低消費電力化という目的に対しても、ＶＬＩＷ方式が最良の方式であるとは必ずしも言えない。

【００２４】

【発明が解決しようとする課題】

以下３点が本発明によって解決しようとする課題である。

(1) 上記動的命令変換部２０３における命令変換及び最適化処理のオーバヘッドを削減する。

（2）異種プロセッサ用プログラムの先読み処理を他の解釈実行処理、命令変換・最適化処理と並列に行い、プログラム処理の性能を向上する。

（3）変換後命令列の高速処理、プロセッサの高速周波数動作及びプロセッサ低消費電力化をＶＬＩＷ方式よりも効果的に実現する。

【００２５】

【課題を解決するための手段】

上記課題を解決するために、本発明の動的命令変換機能を有するプロセッサシステムは、異種ハードウェアプラットフォーム向けの命令バイナリコードプログラムを動的に自身の命令バイナリコードに変換しながらプログラム実行を行う際、当該異種ハードウェアプラットフォーム向けの命令バイナリコードプログラムから１命令づつ読み出し、該１命令づつをソフトウェアを介して解釈実行する処理流と、前記１命令づつ必要に応じて自身の命令バイナリコードに変換し、前記１命令づつ蓄積し、蓄積された命令バイナリコード列を必要に応じて最適化する処理流を独立させてこれらを並列に処理することを特徴とする。

【００２６】

さらに、上記命令バイナリコード列の最適化において、繰り返し処理や手続き呼出し処理などを並列に実行出来るように複数処理流を生成するように新たな命令バイナリコード列を構成し、上記解釈実行する処理流及び上記最適化する処理流とは別に、前記異なるハードウェアプラットフォーム向けの命令バイナリコードプログラムをキャッシュメモリに先読みする処理流を独立させ、上記解釈実行する処理流及び上記最適化する処理流と並列に処理することを特徴とする。

【００２７】

また、上記最適化する処理流が所定の単位の命令バイナリコード列最適化を完了する毎に、前記最適化した命令バイナリコード列をその最適化完了時点で上記解釈実行する処理流が実行している命令コードと入れ替え、前記解釈実行する処理流は異なるハードウェアプラットフォーム向けの命令バイナリコードプログラムの各命令解釈実行時に前記１命令に対応する最適化変換後命令バイナリコード列が存在する場合、前記最適化変換後命令バイナリコード列を実行する機構を有し、さらには、上記複数処理流を効率良く並列処理するために、１個のＬＳＩチ

ップに複数のマイクロプロセッサを実装するチップマルチプロセッサ形式、または、１個の命令実行制御部で同時に複数の処理流を実行する形式で構成される。

【００２８】

さらに、本発明は少なくとも１つの処理流から成る動的命令変換機能を有するプロセッサシステムを備え、前記少なくとも１つの処理流は異種ハードウエアにて実行されるバイナリーコードプログラムを構成する複数の命令を順次先読みし、共有メモリに格納する処理流１と、前記共有メモリに格納された前記複数の命令を並列して同時に解釈実行する処理流２と、前記解釈実行された前記複数の命令の変換を行う処理流３から構成されることを特徴とする動的命令変換機能を有するプロセッサシステムを提供することである。

【００２９】

さらに、本発明は少なくとも１つのマイクロプロセッサ、バス、共有メモリなどを備えた半導体デバイスにおいて、前記少なくとも１つのマイクロプロセッサは、少なくとも１つの処理流を実行するように構成され、前記少なくとも１つの処理流は異種ハードウエアにて実行されるバイナリーコードプログラムを構成する複数の命令を順次先読みし、前記共有メモリに格納する処理流１と、前記共有メモリに格納された前記複数の命令を並列して同時に解釈実行する処理流２と、前記解釈実行された前記複数の命令の変換を行う処理流３から構成され、前記少なくとも１つのマイクロプロセッサは、並列して前記複数の命令を処理しうるように構成されることを特徴とする半導体デバイスを提供することにある。

【００３０】

しかも、本発明はコンピュータに複数の命令の読み出しを行う手順と、該読み出した該複数の命令の内、変換されていない命令に対し変換処理を行う手順と、該変換処理された該命令を実行する手順を並列に実行させるためのバイナリトランスレーションプログラムを提供することにある。

【００３１】

【発明の実施の形態】

本発明の実施の形態を図を用いながら説明する。

【００３２】

図４に本発明に係わる動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の構成を示す。

【００３３】

本実行機構は、実行制御部４０１，元命令列解釈実行部４０２，命令変換・最適化処理部４０３，元命令列先読み処理部４０４の各処理部と，主記憶４０８上のデータ構造体である元命令列４０７，複数の変換後命令列４１０を含む変換後命令列領域４０９，命令対応表４１１等で構成される。

【００３４】

命令対応表４１１は例えば図５のような形で構成される。

【００３５】

命令対応表４１１中の各エントリー５０６は、元命令列中の各命令に対応して用意され、例えば当該命令の元命令列中での先頭からの相対アドレスを用いて一意に識別される。

【００３６】

各エントリー５０６は、変換後命令列有無ビットフィールド５０１，命令実行回数フィールド５０２，その他プロファイル情報フィールド５０３，対応変換後命令列先頭アドレスフィールド５０４，実行中ビットフィールド５０５などで構成される。

【００３７】

変換後命令列有無ビットフィールド５０１は、当該エントリー５０６に対応する元命令に対する変換後命令列４１０が存在するか否かを表示する。当該変換後命令列有無ビットフィールド５０１が、当該エントリー５０６に対応した元命令に対する変換後命令列４１０が存在することを示す内容となっている（例えば「１」を表示）場合、対応変換後命令列先頭アドレスフィールド５０４の値は，当該変換後命令列４１０の主記憶４０８における先頭アドレスである。

【００３８】

逆に、当該変換後命令列有無ビットフィールド５０１が、当該エントリー５０６に対応した元命令に対する変換後命令列４１０が存在しないことを示す内容となっている（例えば「０」を表示）場合、対応変換後命令列先頭アドレスフィー

ルド５０４の値は無効である。

【００３９】

また、命令実行回数フィールド５０２は、当該エントリー５０６に対応する元命令の実行回数を表示する。当該命令実行回数フィールド５０２の値が所定の閾値を越えている場合、当該エントリー５０６に対応した元命令が命令変換・最適化処理部４０３での命令変換及び最適化の対象となる。

【００４０】

さらに、その他プロファイル情報フィールド５０３は、当該エントリー５０６に対応した元命令の実行時に発生した事象をプロファイルとして記録しておくフィールドである。

【００４１】

例えば当該元命令が条件分岐命令であった場合，分岐条件成立／不成立といった情報が当該その他プロファイル情報フィールド５０３に記録される。また、命令変換・最適化処理部４０３での命令変換及び最適化に有益なプロファイル情報なども当該その他プロファイル情報フィールド５０３に記録される。実行中ビットフィールド５０５は、当該エントリー５０６に対応する元命令に対する変換後命令列４１０が存在する場合、又は元命令列解釈実行部４０２が当該変換後命令列４１０を実行中である場合にその旨を示す値（例えば「１」）を表示する。

【００４２】

これ以外の場合、実行中ビットフィールド５０５は、無効値（例えば「０」）を表示している。各フィールドの初期値については、変換後命令列有無ビットフィールド５０１，実行中ビットフィールド５０５が無効値（例えば「０」）であり，命令実行回数フィールド５０２の値は「０」、その他プロファイル情報フィールド５０３も無効値である。

【００４３】

次に、図４に戻って各構成要素の概略動作を説明する。

【００４４】

異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行が開始されると、まず実行制御部４０１が元命令列解釈実行部４０２，命令変換・

最適化処理部４０３，元命令列先読み処理部４０４のそれぞれに対応する３つの独立した処理流を生成する。

【００４５】

元命令列先読み処理部４０４に対応する処理流は、実行する元命令列４０７の先読み処理を行う。

【００４６】

先読みされた元命令列は、元命令列のコピー４０５という形でキャッシュメモリ４０６上に存在することになる。元命令列解釈実行部４０２及び命令変換・最適化処理部４０３が元命令列４０７をアクセスする際には既にキャッシュメモリ４０６上に存在する当該元命令列のコピー４０５をアクセスできる。

【００４７】

元命令列先読み処理部４０４の先読みした元命令が分岐命令であった場合、元命令列先読み処理部４０４は一旦分岐双方向の命令列を一定数先読みしておき，実際に当該分岐命令が元命令列解釈実行部４０２にて処理されるのを待ち、その処理終了後当該分岐命令に対応する命令対応表４１１中のその他プロファイル情報フィールド５０３の値を参照して正しい分岐方向を特定し，以降その方向で元命令列先読みを継続する。

【００４８】

元命令列解釈実行部４０２に対応する処理流は、元命令列中の各命令の解釈実行または当該命令に対応する変換後命令列４１０が存在する場合には当該変換後命令列４１０の直接実行を行う。当該命令を解釈実行するかあるいは当該命令に対応する変換後命令列４１０を直接実行するかは、命令対応表４１１の変換後命令列有無ビットフィールド５０１の値を確認して判断する。

【００４９】

当該命令に対応する変換後命令列有無ビットフィールド５０１の値が対応する元命令に対する変換後命令列４１０が存在しないことを示す内容となっている（例えば「０」を表示）場合、元命令列解釈実行部４０２は当該命令の解釈実行を行う。

【００５０】

逆に，当該変換後命令列有無ビットフィールド５０１の値が対応する元命令に対する変換後命令列４１０が存在することを示す内容となっている（例えば「１」を表示）場合、元命令列解釈実行部４０２は当該命令に対応する対応変換後命令列先頭アドレスフィールド５０４の値から対応する変換後命令列４１０を特定し，当該変換後命令列４１０を直接実行する。

【００５１】

この際、元命令列解釈実行部４０２は、当該変換後命令列４１０の直接実行に先だって当該命令に対応する実行中ビットフィールド５０５の値を有効（例えば「１」）にし、さらに、当該変換後命令列４１０の直接実行終了時に当該実行中ビットフィールド５０５の値を無効（例えば「０」）にする。

【００５２】

また、元命令列解釈実行部４０２は、各元命令に対応する解釈実行または変換後命令列直接実行を実施する毎に当該元命令の実行回数を当該命令に対応する命令実行回数フィールド５０２に書き込み、さらに実行プロファイル情報を当該命令に対応するその他プロファイル情報フィールド５０３に書き込む。

【００５３】

命令変換・最適化処理部４０３に対応する処理流は、元命令列の当該プロセッサシステム自身の命令列への変換及び当該変換後命令列の最適化を行う。

【００５４】

命令変換・最適化処理部４０３は、命令対応表４１１中の各元命令に対応する命令実行回数フィールド５０２を参照し、この値が所定の閾値を越えている場合に、当該元命令を当該プロセッサシステム自身の命令列に変換して変換後命令列４１０を主記憶４０８上の変換後命令列領域４０９に作成し、さらに、前後の元命令に対応する変換後命令列４１０が存在する場合、これらと合わせて最適化を行い，新たな最適化変換後命令列４１０を作成する。

【００５５】

最適化に際しては、前後の元命令含めて対応する命令対応表４１１中のその他プロファイル情報フィールド５０３の値を参照し、最適化のためのヒント情報として使用する。

【００５６】

変換後命令列４１０を作成した命令変換・最適化処理部４０３は、対応する元命令に対する命令対応表４１１中の変換後命令列有無ビットフィールド５０１の値を検査し、これが無効（例えば「０」）を示している場合には、当該変換後命令列有無ビットフィールド５０１の値を有効（例えば「１」）を示す値に書き換え、対応する対応変換後命令列先頭アドレスフィールド５０４に生成した変換後命令列４１０の主記憶４０８上での先頭アドレスを書き込む。

【００５７】

逆に、当該変換後命令列有無ビットフィールド５０１の値が有効（例えば「１」）を示す値であった場合には、さらに対応する実行中ビットフィールド５０５を検査し、これが無効（例えば「０」）を示している場合には、対応する対応変換後命令列先頭アドレスフィールド５０４に生成した変換後命令列４１０の主記憶４０８上での先頭アドレスを書き込み、元々当該対応変換後命令列先頭アドレスフィールド５０４で示されていた変換後命令列４１０のメモリ領域を解放する。

【００５８】

この時、当該実行中ビットフィールド５０５の値が有効（例えば「１」）を示す値であった場合には，当該実行中ビットフィールド５０５の値が無効（例えば「０」）を示す値になるのを待ってから対応する対応変換後命令列先頭アドレスフィールド５０４に生成した変換後命令列４１０の主記憶４０８上での先頭アドレスを書き込み，元々当該対応変換後命令列先頭アドレスフィールド５０４で示されていた変換後命令列４１０のメモリ領域を解放する。

【００５９】

続いて、図１を用いて本発明に係わる動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の処理フローを詳細に説明する。

【００６０】

処理１０１で動的命令変換部による異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行が始まる。続いて処理１０２で処理流が３つに

分割される。

【００６１】

ここで生成された元命令列先読み処理流１０３，元命令列解釈実行処理流１０４，命令変換・最適化処理流１０５の３処理流は並列に動作する。

【００６２】

以降、個々の処理流の処理フローについて説明する。まず、元命令列先読み処理流１０３の処理フローを説明する。処理１０６で元命令列先読み処理が開始される。

【００６３】

次に処理１０７で元命令列の実行順に元命令が先読みされる。処理１０８では先読みした元命令の種類を解釈する。この元命令が分岐命令であるかどうかが処理１０９で判定され、分岐命令であれば処理１１０に、そうでなければ処理１１３に移る。処理１１０では、当該分岐命令の分岐双方向に元命令列の実行順に元命令を先読みする。

【００６４】

次に処理１１１では、当該分岐命令に対応する命令対応表４１１中のその他プロファイル情報フィールド５０３を参照して正しい分岐方向を得る。処理１１２では、正しい分岐方向パスでの先読みをした元命令の種類を解釈する。以降処理１０９に戻って処理が繰り返される。

【００６５】

一方、処理１１３では、次に先読みをすべき領域が元命令列プログラムの領域外であるかどうかを判定する。領域外であれば、処理１１５に移り、元命令列先読み処理を終了する。領域外でなければ、処理１１４に移る。処理１１４では、元命令列解釈実行処理流１０４が終了しているかどうかが判定される。処理が終了していれば処理１１５に移り、元命令列先読み処理を終了する。一方、終了していなければ処理１０７に移り以降処理を繰り返す。

【００６６】

次に、元命令列解釈実行処理流１０４の処理フローを説明する。処理１１６で元命令列解釈実行処理が始まる。

【００６７】

処理１１７では、元命令列の実行順で次の（一番はじめの時は先頭の）元命令に対応する命令対応表４１１中の変換後命令列有無ビットフィールド５０１を参照し、当該元命令に対する変換後命令列４１０が存在するか否かを判定する。変換後命令列４１０が存在すれば、処理１２３に移り、存在しなければ処理１１９に移る。処理１１９では当該元命令を解釈実行し、処理１２２に移る。処理１２３では、当該変換後命令列４１０の実行に先だって当該元命令に対応する命令対応表４１１中の実行中ビットフィールド５０５に当該変換後命令列４１０の実行中を示す値（例えば「１」）を書き込む。

【００６８】

次に処理１１８で当該変換後命令列４１０の直接実行が開始される。当該直接実行処理中、処理１２０でマルチスレッド処理が開始された場合、処理１２１で当該マルチスレッド処理が実施される。当該変換後命令列４１０が最後まで実行されると、処理１３９で当該直接実行処理が終了したことが判定され、処理１２４に移る。処理１２４では、当該元命令に対応する命令対応表４１１中の実行中ビットフィールド５０５に当該変換後命令列４１０を実行中で無いことを示す値（例えば「０」）を書き込む。

【００６９】

次に、処理１２２では、当該元命令に対応する命令対応表４１１中の命令実行回数フィールド５０２，その他プロファイル情報フィールド５０３に実行結果を反映する。続く処理１２５では、次の元命令が存在するかどうかを判定し、存在しなければ処理１２６に移り、元命令列解釈実行処理を終了する。次の元命令が存在する場合には，処理１１７に戻り，以降処理を繰り返す。

【００７０】

続いて、命令変換・最適化処理流１０５の処理フローを説明する。処理１２７で命令変換・最適化処理が開始される。

【００７１】

処理１２８では、命令対応表４１１中の命令実行回数フィールド５０２，その他プロファイル情報フィールド５０３を順次参照する。処理１２９では、当該命

令実行回数フィールド５０２の値が所定の閾値を越えているかどうかが判定され，閾値を越えている場合には、処理１３０に移り、閾値を越えていない場合には処理１２８に戻る。

【００７２】

処理１３０では、当該命令実行回数フィールド５０２の値が所定の閾値を越えている命令対応表４１１中のエントリー５０６に対応する元命令の命令変換を実施し、変換後命令列４１０を主記憶４０８における変換後命令列領域に生成する。

【００７３】

なお、当該変換後命令列４１０生成時には、当該元命令に対応する命令対応表４１１中のその他プロファイル情報フィールド５０３の値を変換後命令列最適化のための情報として使用する。

【００７４】

続いて、処理１３１で、当該元命令の前後の元命令に対応する変換後命令列４１０が存在した場合，それらの変換後命令列４１０を合わせて再度最適化処理を実施する。

【００７５】

最適化処理において、処理１３２でマルチスレッド処理化した方がプログラムの処理効率が上がると判定された場合，処理１３３でマルチスレッド処理化を実施する。

【００７６】

続いて、処理１３４で当該元命令に対応する命令対応表４１１中の変換後命令列有無ビットフィールド５０１に変換後命令列４１０が存在することを示す値（例えば「１」）を書き込み、さらに，同エントリー５０６の対応変換後命令列先頭アドレスフィールド５０４に当該変換後命令列４１０の主記憶４０８における先頭アドレスを書き込む。

【００７７】

処理１３５では、当該元命令に対応する命令対応表４１１中の実行中ビットフィールド５０５を参照し、対応する旧変換後命令列が実行中であるかどうかを判

定する。

【００７８】

実行中であれば実行終了するまで待つ。実行中でなければ、処理１３６にて当該旧変換後命令列４１０のメモリ領域を解放し，廃棄する。

【００７９】

次に処理１３７では、元命令列解釈実行処理が終了しているかどうかを判定し、終了していれば処理１３８に移って命令変換・最適化処理を終了する。元命令列解釈実行処理が終了していなければ、処理１２８に戻って以降処理を繰り返す。

【００８０】

以上が本発明に係わる動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の処理フローである。

【００８１】

ここで、前述される最適化処理とは、変換後命令の並べ替え及び変換後命令数の削減等により命令コードをコンパイラ等のソフトウエアを通して実行コード化した時の実行スピード向上を目指した処理に相当する。

【００８２】

さらに、マルチスレッド処理とは、従来はプログラムの各命令を順次実行するのに対し、各命令を各々のマイクロプロセッサにて並列に同時実行することによる処理能率向上を目指した処理に相当する。

【００８３】

次に、図７，図８を用いて、元命令列先読み処理流１０３，元命令列解釈実行処理流１０４，および命令変換・最適化処理流１０５の相関関係を共有データ構造へのアクセスに着目して説明する。

【００８４】

図７は、キャッシュメモリ４０６上に生成される元命令列のコピー４０５へのアクセスに関する各処理流の相関関係を示す。元命令列のコピー４０５は、元命令列先読み処理流１０３の処理１０７及び処理１１０における元命令先読みによってキャッシュメモリ４０６上に生成される。この元命令列のコピー４０５は，

元命令列解釈実行処理流１０４の処理１１９，および命令変換・最適化処理流１０５の処理１３０における元命令読み出し時にアクセスされる。

【００８５】

図８は、主記憶４０８上に生成される命令対応表４１１のエントリー５０６の変換後命令列有無ビットフィールド５０１，命令実行回数フィールド５０２，その他プロファイル情報フィールド５０３，対応変換後命令列先頭アドレスフィールド５０４，実行中ビットフィールド５０５の各フィールドと、主記憶４０８内の変換後命令列領域４０９上に生成される変換後命令列４１０へのアクセスに関する各処理流の相関関係を示す。

【００８６】

まず第１に変換後命令列有無ビットフィールド５０１は、命令変換・最適化処理流１０５の処理１３４によって値が更新され、元命令列解釈実行処理流１０４の処理１１７で参照される。

【００８７】

次に命令実行回数フィールド５０２は、元命令列解釈実行処理流１０４の処理１２２によって値が更新され、命令変換・最適化処理流１０５の処理１２８から処理１２９までの処理群８０２で参照される。その他プロファイル情報フィールド５０３は、元命令列解釈実行処理流１０４の処理１２２によって値が更新され、元命令列先読み処理流１０３の処理１１１及び命令変換・最適化処理流１０５の処理１３０から処理１３３までの処理群８０１によって参照される。

【００８８】

対応変換後命令列先頭アドレスフィールド５０４は，命令変換・最適化処理流１０５の処理１３４によって値が更新され、元命令列解釈実行処理流１０４の処理１１８から処理１３９までの処理群８０３で参照される。

【００８９】

さらに、実行中ビットフィールド５０５は、元命令列解釈実行処理流１０４の処理１２３及び処理１２４によって値が更新され、命令変換・最適化処理流１０５の処理１３５で参照される。

【００９０】

最後に変換後命令列４１０は、命令変換・最適化処理流１０５の処理１３０から処理１３３までの処理群８０１によって生成され、元命令列解釈実行処理流１０４の処理１１８から処理１３９までの処理群８０３で参照される。

【００９１】

ここで元命令列解釈実行処理流１０４による実行処理中の変換後命令列と命令変換・最適化処理流１０５による変換後命令列の最適化後の新規変換後命令列との間で変換後命令列の入れ替え処理を行う際、排他制御（すなわち、主記憶に内在する共有メモリを処理流１０４及び処理流１０５などが利用する場合、一方の処理流が使用している時、他方の処理流は共有メモリの使用から排除される）が実施される。

【００９２】

ここまででは、本発明に係わる動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の処理方式について説明した。

【００９３】

引き続き以降では、以上のような処理を実行可能なハードウェアプラットフォームについて記述する。

【００９４】

まず図６は、チップマルチプロセッサ６０５の構成例を示す。

【００９５】

本ハードウェアプラットフォームの具体例については、Ｐｒｏｃｅｅｄｉｎｇｓ ｏｆ Ｅｉｇｈｔｈ Ｉｎｔｅｒｎａｔｉｏｎａｌ Ｃｏｎｆｅｒｅｎｃｅ ｏｎ Ａｒｃｈｉｔｅｃｔｕｒａｌ Ｓｕｐｐｏｒｔ ｆｏｒ Ｐｒｏｇｒａｍｍｉｎｇ Ｌａｎｇｕａｇｅｓ ａｎｄ Ｏｐｅｒａｔｉｎｇ Ｓｙｓｔｅｍｓ （ＡＳＰＬＯＳ ＶＩＩＩ）の５８ページから６９ページに掲載されている“Ｄａｔａ Ｓｐｅｃｕｌａｔｉｏｎ Ｓｕｐｐｏｒｔ ｆｏｒ ａ Ｃｈｉｐ Ｍｕｌｔｉｐｒｏｃｅｓｓｏｒ”というタイトルの論文に開示されている。

【００９６】

チップマルチプロセッサ６０５中には、複数のマイクロプロセッサ６０１と、

これらのマイクロプロセッサ６０１を互いに接続する相互結合網６０２，当該相互結合網６０２に接続され、該複数マイクロプロセッサ６０１で共有される共有キャッシュ６０３、および、主記憶インタフェース６０４などが含まれる。

【００９７】

本発明の処理方式の下で発生する複数処理流はそれぞれスレッドと呼ばれ、当該チップマルチプロセッサ６０５中の該複数マイクロプロセッサ６０１に各々のスレッドが個別に割り当てることで、コンパイラ等のソフトウエアを介して上記複数処理流の並列処理が実現される。

【００９８】

次に、図９は、同時複数スレッド実行プロセッサ９０９の構成例を示す。

【００９９】

本ハードウェアプラットフォームの具体例については、ＩＥＥＥ Ｍｉｃｒｏ誌１９９７年９－１０月号１２ページから１９ページに掲載されている“Ｓｉｍｕｌｔａｎｅｏｕｓ Ｍｕｌｔｉｔｈｒｅａｄｉｎｇ：Ａ Ｐｌａｔｆｏｒｍ ｆｏｒ Ｎｅｘｔ－Ｇｅｎｅｒａｔｉｏｎ Ｐｒｏｃｅｓｓｏｒｓ”というタイトルの論文に開示されている。

【０１００】

同時複数スレッド実行プロセッサ９０９は、命令キャッシュ９０１，複数の命令フェッチ部９０２（命令フェッチ部９０２－１から命令フェッチ部９０２－ｎ），命令選択合成部９０３，命令デコード部９０４，命令実行部９０５，複数のレジスタセット９０６（レジスタセット９０６－１からレジスタセット９０６－ｎ），主記憶インタフェース９０７，データキャッシュ９０８などからなる。

【０１０１】

このうち、命令キャッシュ９０１，命令デコード部９０４，命令実行部９０５，主記憶インタフェース９０７，データキャッシュ９０８は通常のマイクロプロセッサのものと基本的に同じである。

【０１０２】

特徴的なのは、複数の命令フェッチ部９０２（命令フェッチ部９０２－１から命令フェッチ部９０２－ｎ），命令選択合成部９０３，及び複数のレジスタセッ

ト９０６（レジスタセット９０６－１からレジスタセット９０６－ｎ）の各部である。複数の命令フェッチ部９０２（命令フェッチ部９０２－１から命令フェッチ部９０２－ｎ），及び複数のレジスタセット９０６（レジスタセット９０６－１からレジスタセット９０６－ｎ）は，それぞれ本発明の同時複数スレッド実行プロセッサ９０９で同時に処理するスレッド毎に１つずつ割り当てられる。

【０１０３】

命令選択合成部９０３は、各時点での各スレッドの処理状況に応じて命令を取り出す命令フェッチ部９０２を制限し、制限した当該命令フェッチ部９０２が持つ実行可能命令候補の中から同時実行可能な組合せで複数命令を選択し，命令デコード部９０４に引き渡す。

【０１０４】

本発明の処理方式の下で発生する複数処理流をそれぞれスレッドとして，当該同時複数スレッド実行プロセッサ９０９中の該命令フェッチ部９０２（命令フェッチ部９０２－１から命令フェッチ部９０２－ｎ），及び該レジスタセット９０６（レジスタセット９０６－１からレジスタセット９０６－ｎ）の組に個別に割り当てることで、上記複数処理流の並列処理が実現される。

【０１０５】

以上が、本発明に係わる実施の形態である．

【０１０６】

【発明の効果】

本発明によって、異種プロセッサ用プログラムを動的に自身の命令列に変換しながらプログラム実行を行う際、命令変換及び最適化処理のオーバヘッドが削減できる。

【０１０７】

さらには、異種プロセッサ用プログラムの先読み処理を他の解釈実行処理及び命令変換・最適化処理と並列に行うことで同プログラム処理の性能を向上できる。

【０１０８】

また、特に、チップマルチプロセッサ方式との組み合わせによって、変換後命

令列の高速処理，プロセッサの高速周波数動作，及び低消費電力化を実現できる。

【図面の簡単な説明】

【図１】

本発明に係わる動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の処理フローを示す図。

【図２】

従来技術での動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の構成を示す図。

【図３】

従来技術での動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の処理フローを示す図。

【図４】

本発明に係わる動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構の構成を示す図。

【図５】

本発明に係わる動的命令変換機能を含んだ異種ハードウェアプラットフォーム向け命令バイナリコードプログラム実行機構が使用する命令対応表の構成を示す図。

【図６】

従来技術のチップマルチプロセッサの構成例を示す図。

【図７】

本発明に係わるキャッシュメモリ上元命令列のコピーを介した各処理流間の相互関係を示す図。

【図８】

本発明に係わる主記憶上命令対応表及び変換後命令列領域を介した各処理流間の相互関係を示す図。

【図９】

従来技術の同時複数スレッド実行プロセッサの構成例を示す図。

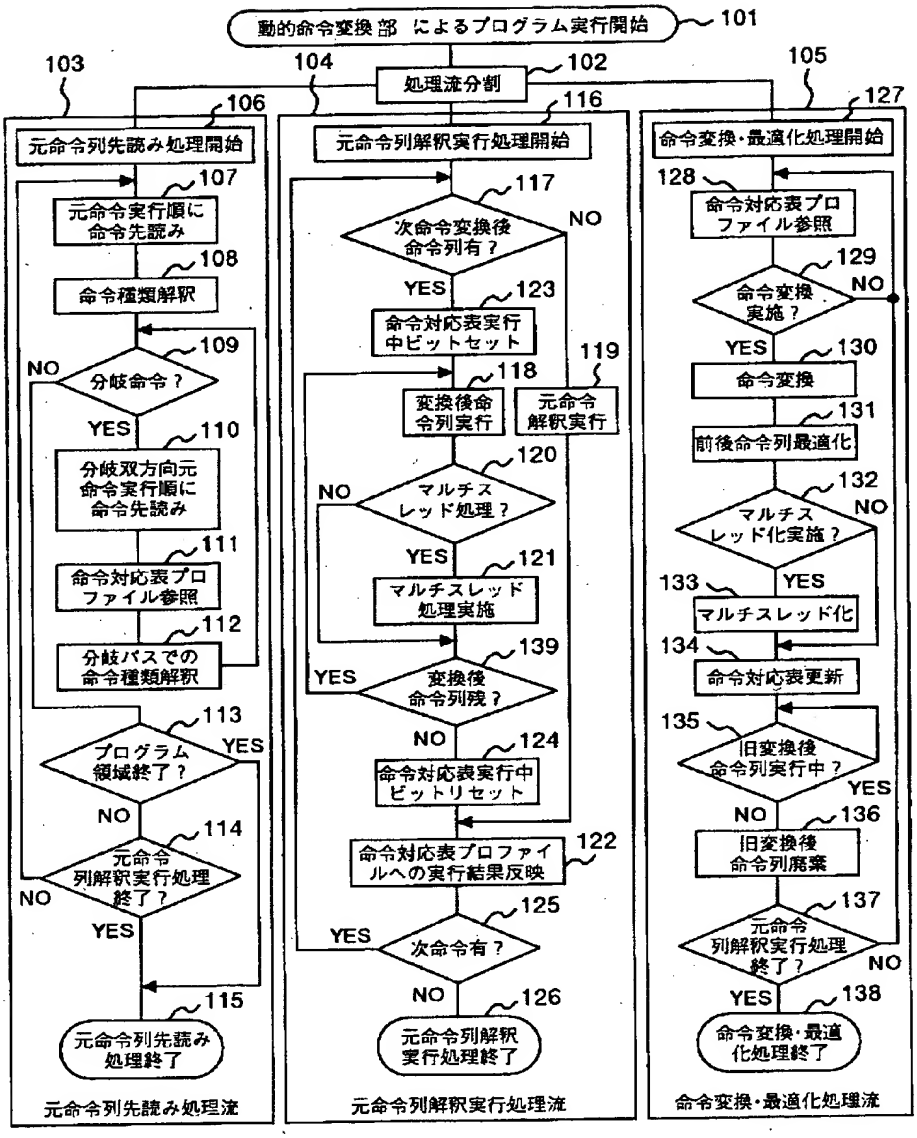
【符号の説明】

１０３…元命令列先読み処理流，１０４…元命令列解釈実行処理流，１０５…命令変換・最適化処理流，２０１…解釈実行部，２０２…実行制御部，２０３…動的命令変換部，２０４…特殊処理エミュレーション部，２０５…プラットフォームＯＳ及びハードウェア，３０６…変換後命令列ブロック，３０８…変換後命令列領域，４０１…実行制御部，４０２…元命令列解釈実行部，４０３…命令変換・最適化処理部，４０４…元命令列先読み処理部，４０５…元命令列のコピー，４０６…キャッシュメモリ，４０７…元命令列，４０８…主記憶，４０９…変換後命令列領域，４１０…変換後命令列，４１１…命令対応表，５０１…変換後命令列有無ビットフィールド，５０２…命令実行回数フィールド，５０３…その他プロファイル情報フィールド，５０４…対応変換後命令列先頭アドレスフィールド，５０５…実行中ビットフィールド，５０６…命令対応表エントリ，６０１…マイクロプロセッサ，６０２…相互結合網，６０３…共有キャッシュ，６０４…主記憶インタフェース，６０５…チップマルチプロセッサ，９０１…命令キャッシュ，９０２－１～９０２－ｎ…命令フェッチ部１～命令フェッチ部Ｎ，９０３…命令選択合成部，９０４…命令デコード部，９０５…命令実行部，９０６－１～９０６－ｎ…レジスタセット１～レジスタセットＮ，９０７…主記憶インタフェース，９０８…データキャッシュ，９０９…同時複数スレッド実行プロセッサ。

【書類名】　図面

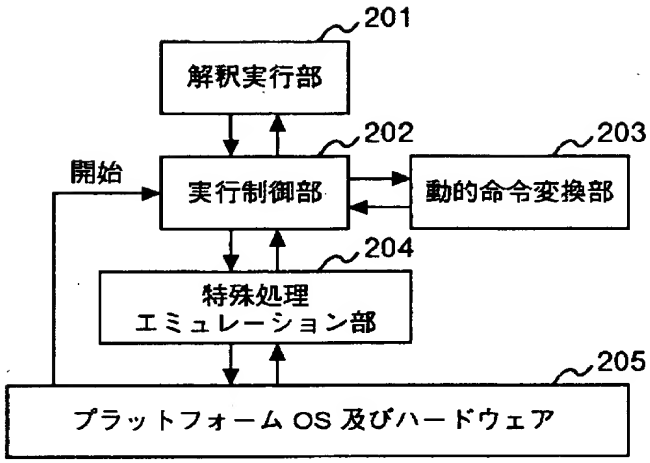
【図１】

図　1


動的命令変換部 によるプログラム実行開始
101
処理流分割
102
103
104
105
元命令列先読み処理開始
106
元命令実行順に命令先読み
107
命令種類解釈
108
分岐命令？
109
NO
YES
分岐双方向元命令実行順に命令先読み
110
命令対応表プロファイル参照
111
分岐パスでの命令種類解釈
112
プログラム領域終了？
113
YES
NO
元命令列解釈実行処理終了？
114
NO
YES
元命令列先読み処理終了
115
元命令列先読み処理流
元命令列解釈実行処理開始
116
次命令変換後命令列有？
117
NO
YES
命令対応表実行中ビットセット
123
変換後命令列実行
118
元命令解釈実行
119
マルチスレッド処理？
120
NO
YES
マルチスレッド処理実施
121
変換後命令列残？
139
YES
NO
命令対応表実行中ビットリセット
124
命令対応表プロファイルへの実行結果反映
122
次命令有？
125
YES
NO
元命令列解釈実行処理終了
126
元命令列解釈実行処理流
命令変換・最適化処理開始
127
命令対応表プロファイル参照
128
命令変換実施？
129
NO
YES
命令変換
130
前後命令列最適化
131
マルチスレッド化実施？
132
NO
YES
マルチスレッド化
133
命令対応表更新
134
旧変換後命令列実行中？
135
YES
NO
旧変換後命令列廃棄
136
元命令列解釈実行処理終了？
137
NO
YES
命令変換・最適化処理終了
138
命令変換・最適化処理流

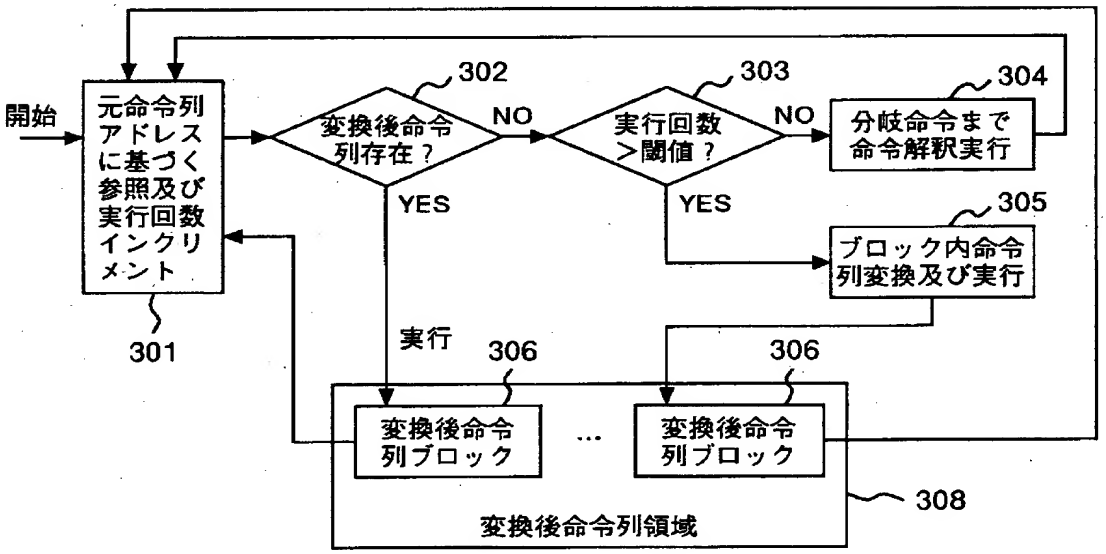
【図２】

図　2


201
解釈実行部
202
203
開始
実行制御部
動的命令変換部
204
特殊処理
エミュレーション部
205
プラットフォーム OS 及びハードウェア

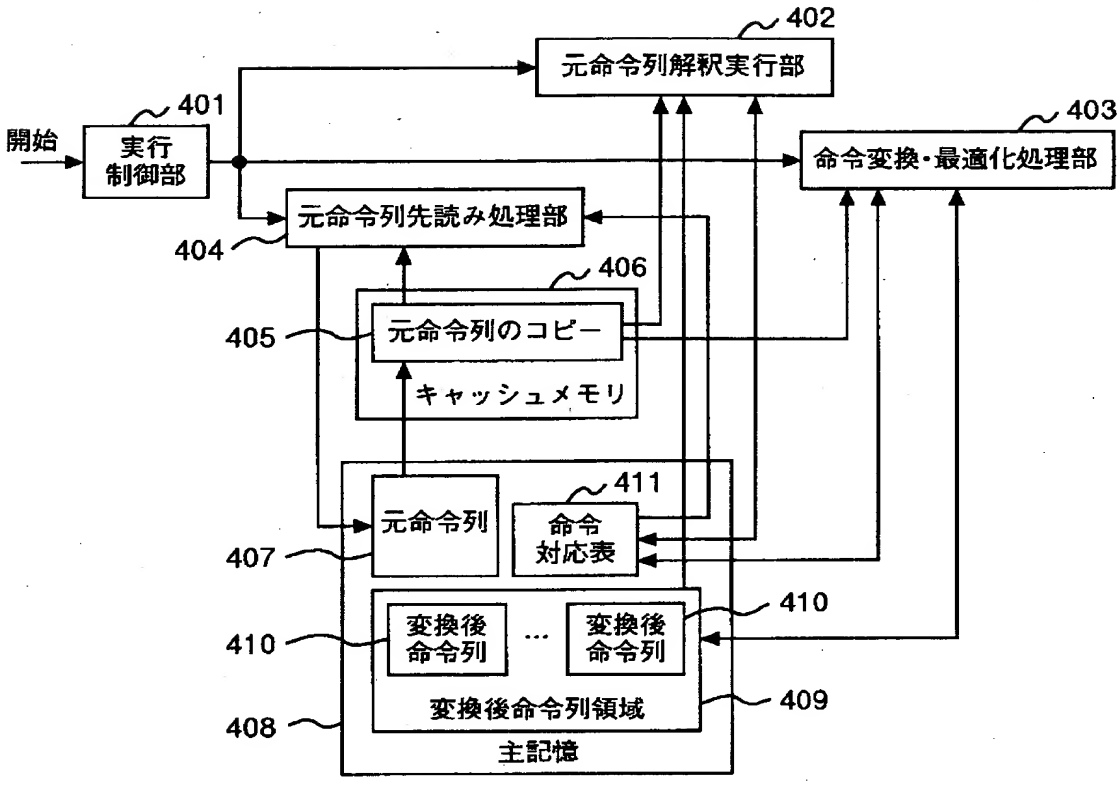
【図３】

図　3


開始
元命令列
アドレス
に基づく
参照及び
実行回数
インクリ
メント
301
302
変換後命令
列存在？
NO
YES
303
実行回数
＞閾値？
NO
YES
304
分岐命令まで
命令解釈実行
305
ブロック内命令
列変換及び実行
実行
306
306
変換後命令
列ブロック
...
変換後命令
列ブロック
308
変換後命令列領域

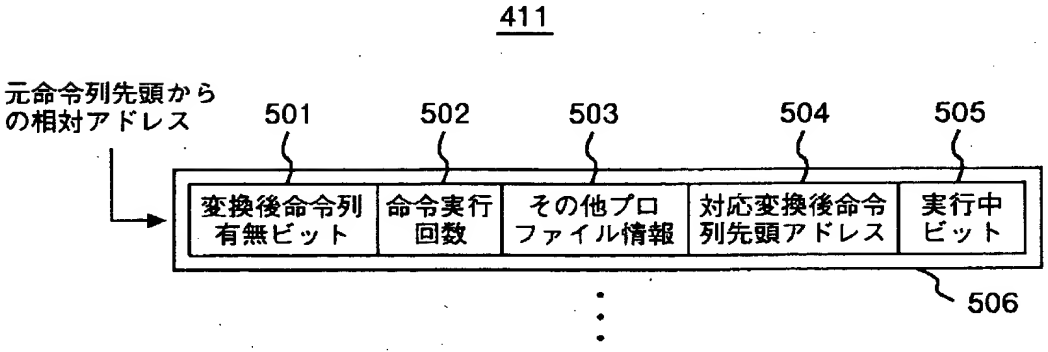
【図４】

図　4


402
元命令列解釈実行部
401
開始
実行
制御部
403
命令変換・最適化処理部
元命令列先読み処理部
404
406
405
元命令列のコピー
キャッシュメモリ
411
元命令列
命令
対応表
407
410
変換後
命令列
…
変換後
命令列
410
変換後命令列領域
409
408
主記憶

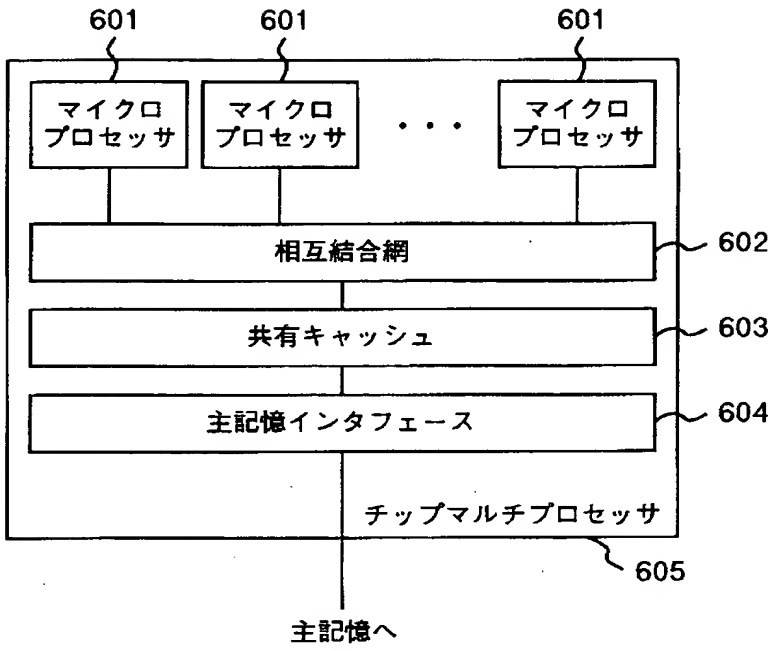
【図５】

図　5


411
元命令列先頭から
の相対アドレス
501
502
503
504
505
変換後命令列
有無ビット
命令実行
回数
その他プロ
ファイル情報
対応変換後命令
列先頭アドレス
実行中
ビット
506

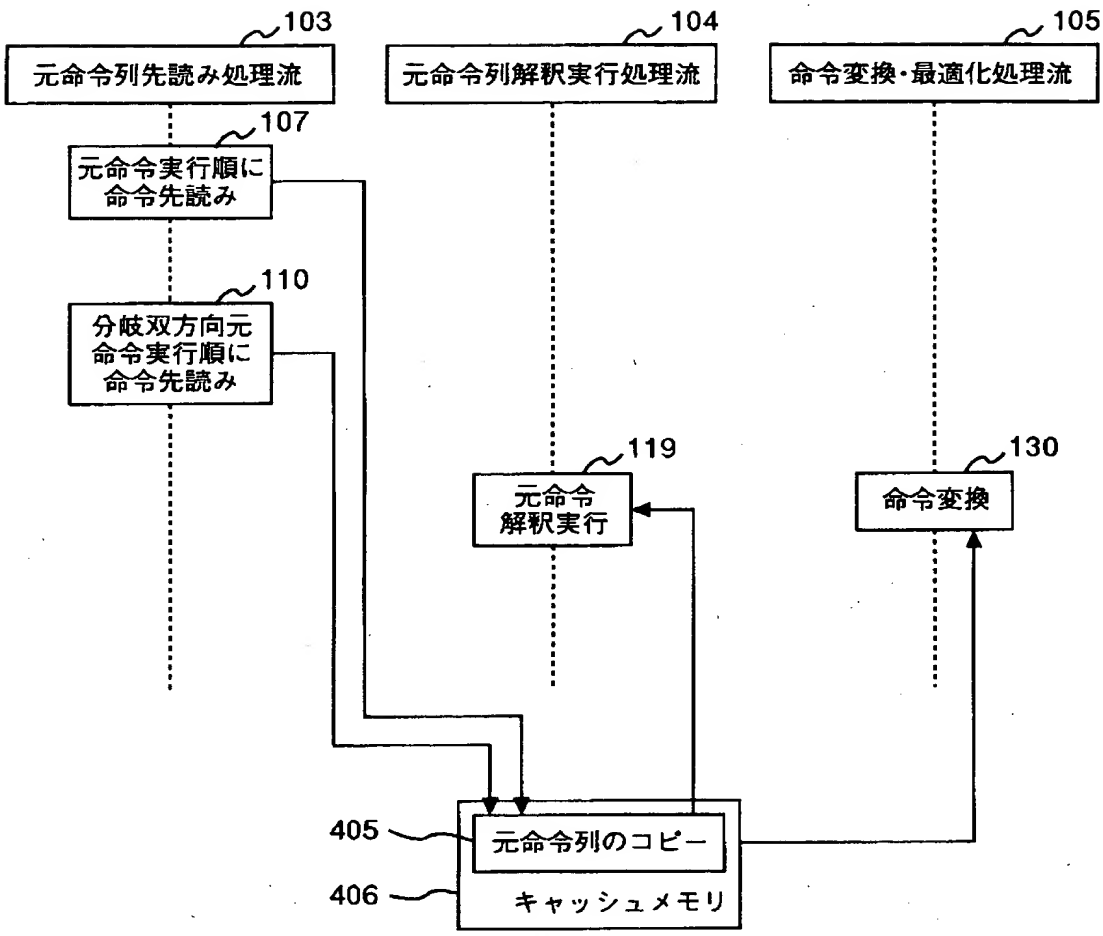
【図６】

図　6


601
601
601
マイクロプロセッサ
マイクロプロセッサ
・・・
マイクロプロセッサ
相互結合網
602
共有キャッシュ
603
主記憶インタフェース
604
チップマルチプロセッサ
605
主記憶へ

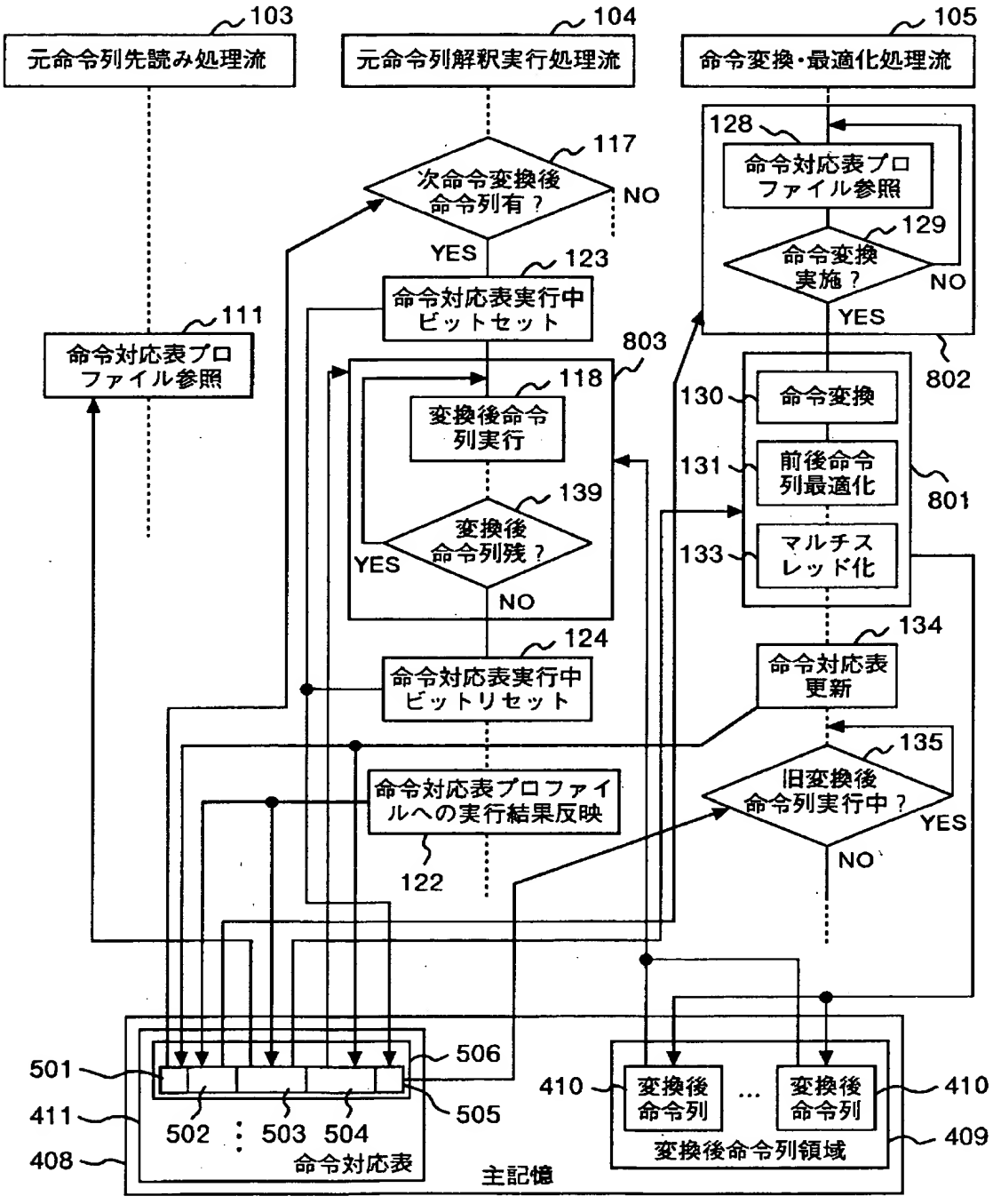
【図７】

図　7


103
元命令列先読み処理流
104
元命令列解釈実行処理流
105
命令変換・最適化処理流
107
元命令実行順に命令先読み
110
分岐双方向元命令実行順に命令先読み
119
元命令解釈実行
130
命令変換
405
元命令列のコピー
406
キャッシュメモリ

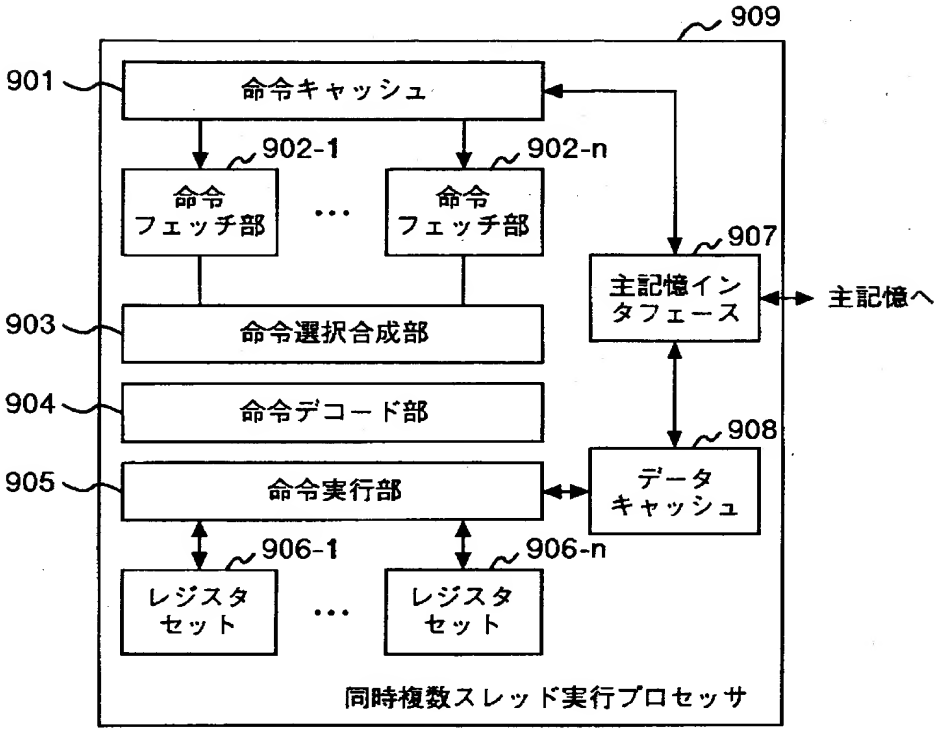
【図８】

図　8


103
元命令列先読み処理流
104
元命令列解釈実行処理流
105
命令変換・最適化処理流
117
次命令変換後命令列有？
NO
YES
123
命令対応表実行中ビットセット
111
命令対応表プロファイル参照
803
118
変換後命令列実行
139
変換後命令列残？
YES
NO
124
命令対応表実行中ビットリセット
命令対応表プロファイルへの実行結果反映
122
128
命令対応表プロファイル参照
129
命令変換実施？
NO
YES
802
130
命令変換
131
前後命令列最適化
801
133
マルチスレッド化
134
命令対応表更新
135
旧変換後命令列実行中？
YES
NO
501
411
408
502
503
504
506
505
命令対応表
主記憶
410
変換後命令列
変換後命令列
410
409
変換後命令列領域

【図９】

図　9


909
901
命令キャッシュ
902-1
902-n
命令
フェッチ部
命令
フェッチ部
907
主記憶イン
タフェース
主記憶へ
903
命令選択合成部
904
命令デコード部
908
905
命令実行部
データ
キャッシュ
906-1
906-n
レジスタ
セット
レジスタ
セット
同時複数スレッド実行プロセッサ

【書類名】　要約書

【要約】

【課題】　異種プロセッサ用命令列実行時の命令変換及び最適化処理オーバヘッドを削減し，同時に，変換後命令列高速処理，プロセッサ高速周波数動作及び低消費電力化を実現する。

【解決手段】　元命令列解釈実行処理流１０４，命令変換・最適化処理流１０５と元命令列先読み処理流１０３を独立させ，チップマルチプロセッサ形式や，１個の命令実行制御部で同時に複数の処理流を実行する形式でプロセッサを構成して上記複数処理流を並列処理する。さらに，命令変換・最適化処理流１０５は，複数処理流を生成する形で変換後命令列を構成し，元命令列解釈実行処理流１０４は，各命令解釈実行時に命令変換・最適化処理流１０５に生成された当該命令に対応する変換後命令列が存在する場合にこれを実行する。

【効果】　上記課題を解決し，さらに元命令列読込み処理オーバヘッドを削減する。

【選択図】　図１

出証特２００１－３０５７０４８

出　願　人　履　歴　情　報

識別番号　　〔０００００５１０８〕

１．変更年月日　　１９９０年　８月３１日
［変更理由］　新規登録
住　所　東京都千代田区神田駿河台４丁目６番地
氏　名　株式会社日立製作所